Some improvements of the spectral learning approach for probabilistic grammatical inference

Mattias Gybels, Francois Denis, Amaury Habrard

JMLR: Workshop and Conference Proceedings 34:1–15, 2014 Proceedings of the 12th International Conference on Grammatical Inference

Some improvements of the spectral learning approach for probabilistic grammatical inference

Mattias Gybels MATTIAS.GYBELS@LIF.UNIV-MRS.FR
François Denis FRANCOIS.DENIS@LIF.UNIV-MRS.FR
Aix Marseille Université, CNRS, LIF UMR 7279, 13288 Marseille Cedex 9, FRANCE

Amaury Habrard AMAURY.HABRARD@UNIV-ST-ETIENNE.FR
Université Jean Monnet de Saint-Etienne, CNRS LaHC UMR 5516, 42000 Saint-Etienne, FRANCE

Editors: Alexander Clark, Makoto Kanazawa and Ryo Yoshinaka

Abstract

Spectral methods propose new and elegant solutions in probabilistic grammatical inference. We propose two ways to improve them. We show how a linear representation, or equivalently a weighted automata, output by the spectral learning algorithm can be taken as an initial point for the Baum Welch algorithm, in order to increase the likelihood of the observation data. Secondly, we show how the inference problem can naturally be expressed in the framework of Structured Low-Rank Approximation. Both ideas are tested on a benchmark extracted from the PAutomaC challenge.

Keywords: Probabilistic grammatical inference, Rational series, Machine Learning, Spectral learning, Structured low-rank approximation, Maximum-Likelihood, Baum-Welch Algorithm.

1. Introduction

One of the main tasks of *probabilistic grammatical inference* consists in learning probabilistic models of *stochastic languages*, i.e. probability distributions, from finite samples of strings or trees. This field has recently made huge advances with the emergence of spectral methods, which offer new and elegant solutions to this problem. The proposed algorithms are indeed consistent and efficient. Many approaches have been proposed to learn HMM or Probabilistic Finite Automata (PFA) (Bailly et al., 2009; Hsu et al., 2009; Bailly et al., 2010; Balle and Mohri, 2012; Balle et al., 2012, 2014), transducers (Balle et al., 2011) or graphical models (Parikh et al., 2011; Luque et al., 2012; Anandkumar et al., 2012; Cohen et al., 2013).

In this paper, we mainly focus on the following problem : given a finite sample of strings S independently and identically drawn (i.i.d.) according to an unknown distribution p, infer an approximation of p in the class of *rational series*. Rational series over a finite alphabet Σ are mappings from Σ^* to $\mathbb{R}$ that can equivalently be defined by means of *weighted automata*, or *linear representations*. More precisely, a series $r : \Sigma^* \mapsto \mathbb{R}$ is rational if there exists an integer k and a tuple $A =< I, (M_x)_{x\in\Sigma}, T >$, where $I, T \in \mathbb{R}^k$ and $M_x \in \mathbb{R}^{k\times k}$ for any $x \in \Sigma$, such that $r(x_1 \dots x_n) = I^\top M_{x_1} \dots M_{x_n} T$. A is called a k-dimensional linear representation of r. The rank of r is the minimal dimension of a linear representation for r.

The spectral learning approach relies on the two following observations :

— the Hankel matrix of a rational series r, i.e. the bi-infinite matrix indexed by $\Sigma^* \times \Sigma^*$ and defined by $H[u, v] = r(uv)$, has a finite rank, equals to the rank of r ;

— a linear representation of r can easily be obtained from the right singular vectors of H.

The spectral learning scheme for probabilistic grammatical inference consists then in building the empirical Hankel matrix H_S from S, performing a singular value decomposition (SVD) of H_S and building a linear representation by using the k-first right singular vectors of H_S, where the rank k is given or estimated by cross-validation or other methods. This main scheme admits several variants.

The spectral approach is consistent, elegant and simple to implement. However it should and can be improved. Indeed, considering the k-first right singular vectors of H_S boils down to perform a rank-k truncated SVD of H_S, i.e. to consider the rank-k matrix $H_{S,k}$ which is the closest to H_S according to the Frobenius norm. But $H_{S,k}$ is not a Hankel matrix - we may have $uv = u'v'$ and $H_{S,k}[u, v] \neq H_{S,k}[u', v']$ - and building a linear representation from the right singular vectors of $H_{S,k}$ takes away from the solution. Another way to express the same thing is to say that the Frobenius norm is maybe not the distance that we should minimize. For example, there is no reason that the output solution maximizes the likelihood of the observations.

Hence, a first simple idea would be to modify the parameters of the solution output by the spectral learning algorithm so as to increase the likelihood of the learning set. Besides the fact that maximizing the likelihood of data is a hard-problem, we face a specific difficulty : the automaton output by the spectral algorithm is not probabilistic – it may contain negative coefficients, it is not normalized and the sum of the values computed on all strings may differ from 1. Therefore, standard ways to increase the likelihood of observation data, such as the Baum-Welch algorithm, cannot be directly applied. In this paper, we show how to transform the output representation into a representation whose negative coefficients can be made arbitrarily close to 0. Thus normalizing such a representation leads to an automaton which is probabilistic and that can be taken as an input by the Baum-Welch algorithm. The underlying idea is that, if the solution output by the spectral learning algorithm is good enough, taking it as an initial point for the Baum Welch algorithm should further improve the solution. This protocol is experimented on a benchmark made of 10 problems extracted from the PAutomaC Challenge (Verwer et al., 2012). The experiments confirm the expected results.

A second idea is to set out precisely from the criteria to be optimized and to design an algorithm able to achieve this optimization task : we look for a rank-k Hankel matrix being the closest to the empirical matrix H_S. This optimization task can be expressed and solved within the Structured Low Rank Approximation framework (SLRA), a bunch of methods aiming at solving constrained low-rank approximation problem (Markovsky, 2008, 2012; Markovsky and Usevich, 2014). However the problem is non convex and the available algorithms and software can only deal with small dimensions matrices. Experiments on the same benchmark demonstrate the potentiality of this approach. However, the current algorithm cannot process high dimensional matrices and the global improvements are quite limited.

The paper is organized as follows. Section 2 introduces the preliminaries on the spectral learning approach and describes the PAutomaC Challenge data. Section 3 describes how the representation output by a spectral learning algorithm can be used as an initialization point for the Baum-Welch algorithm. Section 4 describes the SLRA approach. A conclusion ends the paper.

2. Preliminaries

2.1. Rational series and residuals

Let Σ be a finite alphabet and Σ^* be the set of all strings defined on Σ. Let us denote by $\prec$ the quasi-lexicographical ordering on Σ^* : strings are ordered first by length and then by lexicographical order. Let ϵ denote the empty string and $|w|$ denote the length of w. For any $n \in \mathbb{N}$, let $\Sigma^n = \{w \in \Sigma^* : |w| = n\}$ and $\Sigma^{\leq n} = \{w \in \Sigma^* : |w| \leq n\}$. For any $w \in \Sigma^*$, let $\text{pref}(w) = \{u \in \Sigma^* : \exists v \in \Sigma^*,\ uv = w\}$ and $\text{suff}(w) = \{u \in \Sigma^* : \exists v \in \Sigma^*,\ vu = w\}$. A set of strings P is a *prefix* set if and only if $\forall w \in P, \text{pref}(w) \subseteq P$. For any $S \subseteq \Sigma^*$, let $\text{front}(S) = \{ux \in \Sigma^* : u \in S, x \in \Sigma, ux \notin S\}$. A *series* r defined over Σ is a mapping $r : \Sigma^* \mapsto \mathbb{R}$. A series r is *convergent* if the sequence $r(\Sigma^{\leq n}) = \sum_{w \in \Sigma^{\leq n}} r(w)$ is convergent : the limit is denoted by $r(\Sigma^*)$. A series r is *absolutely convergent* if the series $|r|$ defined by $|r|(w) = |r(w)|$ is convergent. The set of all absolutely convergent series is denoted by $l_1(\Sigma^*)$. A *stochastic language* p is a probability distribution defined on Σ^*, ie a series that only takes non negative values and that converges to 1.

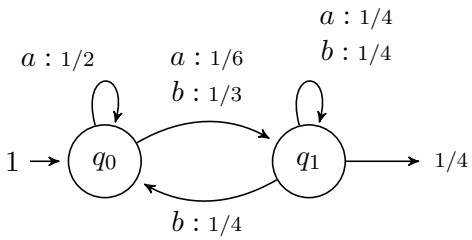
A series r over Σ is *rational* if there exists an integer $k \geq 1$, $I, T \in \mathbb{R}^k$ and matrices $M_x \in \mathbb{R}^{k \times k}$ for every $x \in \Sigma$, such that for all $u = x_1 x_2 \dots x_m \in \Sigma^*$,

$$r(u) = I^T M_u T = I^T M_{x_1} M_{x_2} \dots M_{x_m} T.$$

The triplet $\langle I, (M_x)_{x \in \Sigma}, T \rangle$ is called a k-dimensional *linear representation* of r. The *rank* of a rational series r is the minimal dimension of a linear representation of r. Linear representation are equivalent to weighted automata where each coordinate corresponds to a state, I provides the initial weights, T, the terminal weights and each matrix M_x, the x-labeled transition weights. If $r \in l_1(\Sigma^*)$ is rational and admits the minimal linear representation $\langle I, (M_x)_{x \in \Sigma}, T \rangle$, then $r(\Sigma^*) = I^T (I_d - M_\Sigma)^{-1} T$ where I_d is the rank d identity matrix and $M_\Sigma = \sum_{x \in \Sigma} M_x$. A *probabilistic automaton* (PA) can be defined by a linear representation whose coefficients are all non negative and such that $T^\top I = 1$, $I_d - M_\Sigma$ is invertible and $(I_d - M_\Sigma)^{-1} T = \mathbf{1}$ where $\mathbf{1} = (1, \dots, 1) \in \mathbb{R}^k$.

A k-dimensional linear representation $\langle I, (M_x)_{x \in \Sigma}, T \rangle$ is *prefix* if $I = (1, 0, \dots, 0)^\top \in \mathbb{R}^k$ and if there exists a prefix set $P = \{u_1, u_2, \dots, u_k\}$ such that $\forall i \in [k], \forall x \in \Sigma,\ \exists j \in [k]$ s.t. $u_i x = u_j \Rightarrow \forall k \neq j, M_x[i, k] = 0$.

For any series $r \in l_1(\Sigma^*)$ and any $u \in \Sigma^*$, let $\dot{u}r$ be the series defined by $\dot{u}r(v) = r(uv)$: it is called the *residual* of r wrt u. If $\langle I, (M_x)_{x \in \Sigma}, T \rangle$ is a linear representation of r, then $\langle M_u^T I, (M_x)_{x \in \Sigma}, T \rangle$ is a linear representation of $\dot{u}r$. Let $res(r)$ be the set of all residuals of r and $\mathcal{V}_r$ be the subspace of $l_1(\Sigma^*)$ spanned by $res(r)$. The set of all series $s \in \mathcal{V}_r$ for which $s(\Sigma^*) = 1$ forms a hyperplane $\mathcal{H}_r$ of $\mathcal{V}_r$. The projection of any residual $\dot{u}r$ s.t. $r(u\Sigma^*) \neq 0$ on $\mathcal{H}_r$ is denoted by $u^{-1}r$ and defined by $\forall v \in \Sigma^*$, $u^{-1}r(v) = \frac{r(uv)}{r(u\Sigma^*)}$.



$$I = \begin{bmatrix}1\\0\end{bmatrix} \quad T = \begin{bmatrix}0\\1/4\end{bmatrix}$$
$$M_a = \begin{bmatrix}1/2 & 1/6\\0 & 1/4\end{bmatrix} \quad M_b = \begin{bmatrix}0 & 1/3\\1/4 & 1/4\end{bmatrix}$$

Figure 1: A probabilistic automaton and the equivalent linear representation.

2.2. Hankel matrix and spectral algorithm

The *Hankel matrix* H_r of a series r is a bi-infinite matrix whose rows and columns are indexed by Σ^* and defined by $\forall u, v \in \Sigma^*$, $H_r[u, v] = r(uv)$.

$$H_r = \begin{array}{c} \\ \epsilon \\ a \\ b \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccccc} \epsilon & a & b & aa & \dots \end{array} \\ \begin{pmatrix} r(\epsilon) & r(a) & r(b) & r(aa) & \dots \\ r(a) & r(aa) & r(ab) & r(aaa) & \dots \\ r(b) & r(ba) & r(bb) & r(baa) & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \end{array}$$

A series r is rational iff the rank of its Hankel matrix H_r is finite, and in this case both ranks coincide. Let r be a rank-k rational series and let $H_r = LDR^\top$ be a singular values decomposition of H_r with $L, R \in \mathbb{R}^{\infty \times k}$ the right and left singular vectors of H_r, respectively and $D \in \mathbb{R}^{k \times k}$ be the diagonal matrix containing its singular values. The spectral approach in probabilistic grammatical inference is based on the following result :

— let $I = R[1, :]$ *be the first row of the matrix* R
— let $M_x = R^T T_x R$ *where* T_x *is the constant matrix defined by* $T_x[u, v] = \delta_{v=ux}$,
— let $T = R^T H[1, :]^T$

then $\langle I, (M_x)_{x\in\Sigma}, T\rangle$ is a linear representation of r.

The most basic learning algorithm derived from this result consists in plugging-in the empirical Hankel matrix H_S instead of the unknown matrix H_r.

Let S be a training set of strings i.i.d. from an unknown distribution p. Let H_S be the empirical Hankel matrix built from S defined by $H_S[u, v] = p_S(uv)$ where p_S is the empirical distribution associated with S. Let $U = \cup_{w\in S}\text{pref}(w)$ and $V = \cup_{w\in S}\text{suff}(w)$: H_S is indexed by U and V. Let $H_S \approx L_k D_k R_k^\top$, a rank-$k$ truncated SVD of H_S : L_k (resp. R_k) is composed of the k-first left (resp. right) singular vectors and D_k is the diagonal matrix composed of the k first singular values of H_S. The matrices I, $(M_x)_{x\in\Sigma}$ and T are then computed as above, using R_k instead of R and using $H_S[1, :]$ instead of $H_r[1, :]$. Note that the choice of k is an important parameter of the spectral algorithm (Kulesza et al., 2014) but it will not be discussed in this paper. Several variants of this basic algorithm have been studied (Balle et al., 2012, 2014; Denis et al., 2014).

2.3. PAutomaC

The Probabilistic Automata learning Competition (PAutomaC[1]), proposed in sidelines of the ICGI 2012 conference, is about the problem of learning probabilistic distributions from strings drawn from finite state automata.

The competition offers to solve a set of 48 problems provided in the forme of target models. These models can be of the four following types : Markov Chains, Determinist Probabilistic Finite Automata, Hidden Markov Models and Probabilistic Finite Automata. The participants have acces to training sets of strings i.i.d. from the target models and to test sets. The evaluation process involves making good approximations of the probabilities of the strings from the test set. The quality of the approximation is evaluated by the perplexity criterion defined as follows :

$$2^{-\left(\sum_{w \in TestSet} p(w) * \log(\hat{p}(w))\right)}$$

where p is the target series and $\hat{p}$ is the candidate series that is evaluated. Because models learned using spectral methods can return negative values, a *trigram* trained on S will be used for strings for which $\hat{p}$ outputs a negative value. In this paper, we focus on the 10 problems for which the rank of the target is the smallest, in order to use classical SVD decomposition software and the software from (Markovsky, 2012) to compute low-rank approximation on small matrices as explained in Section 4. More details about PAutomaC can be found in (Verwer et al., 2012).

3. Using spectral learning as Baum-Welch algorithm initialization

A classical and consistent way to solve our inference problem consists in finding a low-rank linear representation that maximizes the likelihood of the learning set. However, it is well known that this problem is NP-hard and approximation algorithms should be used in practical cases. The Baum-Welch algorithm (BW), a straightforward adaptation of the expectation-maximization (EM) algorithm for the framework of PAs, iteratively modifies the parameters of an initial model in order to increase the likelihood of the observation data. The Baum-Welch algorithm converges to a local optimum. A common strategy consists in running BW on a large number of randomly generated initial models and keeping the model which maximizes the likelihood of the learning set. However, the underlying space is strongly non-convex and this strategy is not efficient in high dimension.

On the other hand, spectral learning algorithms are designed to provide low-rank linear representations which are proved to be close to the target. But, as the underlying optimization criterion is of an algebraic nature, spectral learning algorithms are not tailored to maximize the likelihood of the observation data. A sound strategy should be to take the model r output by a spectral learning algorithm as an initialization model for the Baum-Welch algorithm. However, the representation of r has some characteristics that make the Baum-Welch algorithm impossible to be applied on : (i) it may include negative coefficients ; (ii) it may be not normalized, i.e. its components may be far from satisfying the syntactical properties of a PA : $\sum_i I[i] = 1$ and $T[i] + \sum_{j,x} M_x[i,j] = 1$ for any index j.

1. `http://ai.cs.umbc.edu/icgi2012/challenge/Pautomac/`

We show below that given a minimal linear representation of a rational stochastic language p, we can build an equivalent prefix linear representation which is normalized. Moreover, we show that we can build equivalent normalized prefix linear representation where the magnitude of the negative coefficients is arbitrarily small. These techniques will allow us to compute normalized prefix linear representation arbitrarily close from r and on which, the Baum-Welch algorithm could be applied.

3.1. Normalized prefix linear representation and negative coefficients reduction

Let $A =< I, (M_x)_{x\in\Sigma}, T >$ be a rank-k minimum linear representation that computes the stochastic language p and let $P = \{u_1, \ldots, u_k\} \subset \Sigma^*$ be a prefix set where $i \leq j \Rightarrow u_i \prec u_j$ and such that $\{\dot{u}p : u \in P\}$ is a basis of $\mathcal{V}_p$. Then, let $A' =< I', (M'_x)_{x\in\Sigma}, T' >$ be the linear representation defined by

1. $I' = (1, 0, \ldots, 0)^\top \in \mathbb{R}^k$,
2. $\forall i \in [k]$, $T'[i] = \frac{p(u_i)}{p(u_i\Sigma^*)}$,
3. $\forall i, j \in [k], \forall x \in \Sigma$,
   (a) $u_i x = u_j \Rightarrow M'_x[i, j] = \frac{p(u_i x\Sigma^*)}{p(u_i\Sigma^*)}$,
   (b) $u_i x \in front(P) \Rightarrow M'_x[i, j] = \alpha^j_{i,x} \frac{p(u_j\Sigma^*)}{p(u_i\Sigma^*)}$, where $\overline{\dot{u_i x}}p = \sum_{j=1}^k \alpha^j_{i,x} \overline{\dot{u_j}}p$

Then, A' is a normalized prefix linear representation equivalent to A. See an example in Figure 2. Note that the conditions on P implies that $u_1 = \epsilon$, that $p(u_i\Sigma^*) \neq 0$ for any $i \in [k]$ and that the coefficients $\alpha^j_{i,x}$ are uniquely determined.

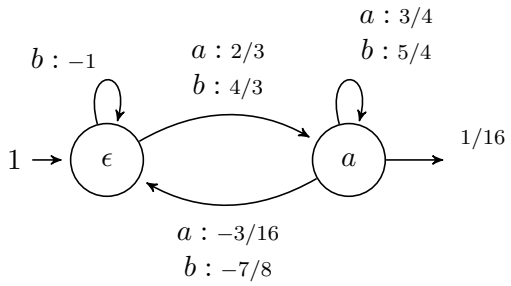


Figure 2: Normalized prefix representation of automaton from Figure 1 for $P = \{\epsilon, a\}$.

If we relax the conditions in such a way that the set $\{\dot{u}p : u \in P\}$ spans $\mathcal{V}_p$ but may be not linearly independent, then the equations $\overline{\dot{u_i x}}p = \sum_{j=1}^k \alpha^j_{i,x} \overline{\dot{u_j}}p$ may not determine the coefficients $\alpha^j_{i,x}$. Additional constraints can be added in order to minimize the magnitude of negative coefficients. From now on, we will replace the conditions $\overline{\dot{u_i x}}p = \sum_{j=1}^k \alpha^j_{i,x} \overline{\dot{u_j}}p$ with

$$\overline{\dot{u_i x}}p = \sum_{j=1}^{k} \alpha^j_{i,x} \overline{\dot{u_j}}p \text{ and } \sum_{j=1}^{k} |\alpha^j_{i,x}| p(u_j\Sigma^*) \text{ minimal}$$

which is equivalent to demand that $\sum_{M'[i,j]<0} |M'[i,j]|$ be minimal since $\sum_{j=1}^{k} \alpha_{i,x}^{j} p(u_j\Sigma^*) = p(u_i x\Sigma^*)$ is a constant.

Not all rational stochastic languages can be computed by a PA (Dharmadhikari, 1963; Denis and Esposito, 2008) (see Fig. 3).

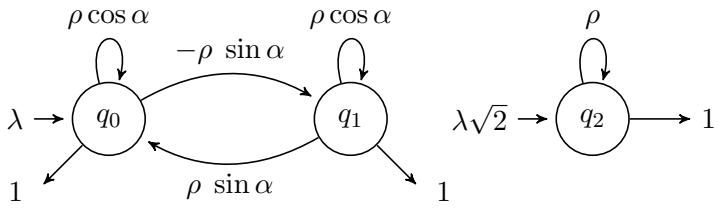


FIGURE 3: For any $\alpha \in \mathbb{R}$ and $0 < \rho < 1$, there exists $\lambda > 0$ such that the automaton defines a stochastic language, which can be computed by a PA iff $\alpha/\pi \in \mathbb{Q}$.

However, it has been shown in (Bailly and Denis, 2011) that any rational stochastic language admits a normalized prefix linear representation where the total magnitude of the negative coefficients is arbitrarily small. See an example on Fig. 4.

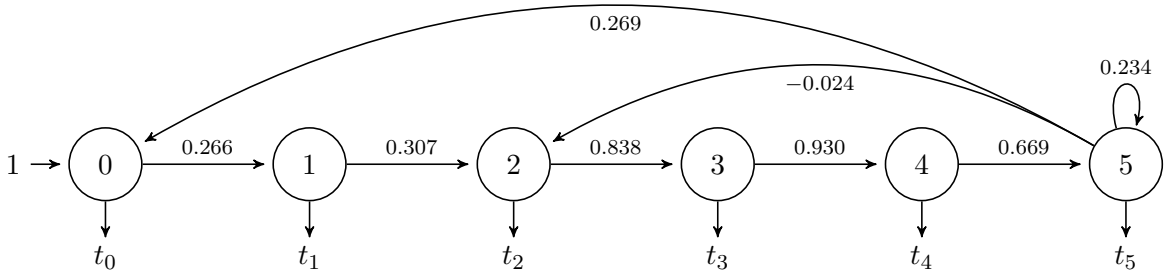


FIGURE 4: Given the automaton described in Fig. 3, with $\rho = 0.5, \alpha = \arccos 0.6$ and $\lambda = 0.3040..$, it defines a stochastic language. Let A_n be the n-dimensional normalized prefix linear representation which minimizes the magnitude of the sum D_n of the negative weights. We have $D_3 = -1.793$, $D_4 = -0.358$, $D_5 = -0.082$ and $D_6 = -0.024$. The figure shows A_6 and $\forall i \in [n]$, $t_i = 1 - \sum_{x,j} M_x[i,j]$.

3.2. Baum-Welch initialization

The spectral algorithm outputs linear representations that converge to the target distribution in ℓ_1 norm (Bailly, 2011; Hsu et al., 2009) : so, it can be assumed that if the learning set is big enough, the output series r is absolutely convergent. But r is not a stochastic language : we can not assume that $r(\Sigma^*) = 1$ neither that $\forall u \in \Sigma^*$, $r(u) \geq 0$. We want to show that our approach - choosing r as the initial model for BW - improves the result of

the spectral learning algorithm when the output representation is sufficiently close to the target.

Experiments on PAutomaC benchmark show that for most problems, the representation output by the spectral algorithm is such that $r(\Sigma^*)$ is close to 1 and $\sum_{u \in S : r(u) < 0} r(u)$ is small (cf. Figure 1). Thereby, given the rank-k series r output by the spectral learning algorithm,

— We consider the series $r' = \frac{r}{r(\Sigma^*)}$.

— We choose an initial prefix set P of size k which maximizes the volume $|Det(\{\dot{u}r : u \in P\})|$ and such that $\forall u \in P$, $r'(u\Sigma^*) > 0$.

— We compute the normalized prefix linear representation $A_0 = \langle I^{(0)}, (M_x^{(0)})_{x \in \Sigma}, T^{(0)} \rangle$ of r', according to P.

We then define a sequence of linear representations A_m by :

— Let $m = 0$.

— While there exists $i, j \in [k]$ and $x \in \Sigma$ such that $M_x^{(m)}[i,j] < 0$

— let $i \in [k]$ and $x \in \Sigma$ such that $\sum_{j ; M_x^{(m)}[i,j] < 0} |M_x^{(m)}[i,j]|$ is maximal ; let $u \in P$ be the string corresponding to i ; $P = P \cup \{ux\}$;

— let $m = m + 1$ and let A_m be the normalized prefix linear representation of r' associated with P.

Every linear representation A_m is equivalent to r' and the global magnitude of the negative coefficients tends to 0 as m increases. A PA B_m is computed from each A_m by applying the following normalization procedure :

$$M_x[i,j] = \frac{|M_x[i,j]|}{\sum_k |M_x[i,k]| + |T[i]|} \text{ and } T[i] = \frac{|T[i]|}{\sum_k |M_x[i,k]| + |T[i]|}.$$

If the global magnitude of the negative coefficients of A_m is small and if r is close to the target, the series computed by the PA B_m should be also close to the target. It can serve as an initial point for the Baum-Welch algorithm.

3.3. Experimentation on PAutomaC problems

In this section, we apply the method described in section 3.2 on 10 problems extracted from the PAutomaC challenge. Some descriptive elements can be found in Table 1. Each problem proposes a training set S (of 2×10^4 or 1×10^5 strings) drawn according to a target model p and a test set (of 1×10^3 strings) on which the perplexity of a candidate series will be evaluated.

Let H_S be the empirical Hankel matrix of S indexed by the sets U and V with $|U| \leq 3000$ (we constrain $|U|$ in order to allow the use of standard SVD using *NumPy* and *SciPy* implementations). Let $\hat{A}^{(k)}$ be the rank-k representation learned from H_S by using the spectral algorithm. A random sub-sampling validation – taking 25% of S as validation set – is performed to determine the best rank in term of likelihood : i.e. $k^* = \underset{k}{argmax}\ L(\hat{A}^{(k)})$, where $L(\hat{A}^{(k)})$ denotes the likelihood of the model $\hat{A}^{(k)}$ computed on the validation set. Let $\hat{p}$ be the series computed by $\hat{A}^{(k^*)}$. Results are detailed in Table 1.

Then, we computed a sequence of m linear representations $B_m^{(r)}$ for any given rank r and any $m \in [0, 10]$ that can be used as starting point for the Baum-Welch algorithm. We denote

$C_m^{(k)}$ the linear representations obtained by applying the Baum-Welch algorithm on $B_m^{(k)}$. Let $(k^*, m^*) = \underset{k,m}{argmax}\ L(\hat{C}_m^{(k)})$, where $L(\hat{C}_m^{(k)})$ is computed on a validation set. Results are detailed in Table 2.

Problem number	4	7	12	20	24	30	31	33	39	42
$\|\Sigma\|$	4	13	13	18	5	10	5	15	14	9
Target series *rank*	10	12	12	8	5	9	12	6	6	6
perplexity target	80.818	51.224	21.655	90.972	38.729	22.926	41.214	31.865	10.002	16.004
k^*	11	12	13	9	5	10	13	7	7	7
perplexity $\hat{A}^{(k^*)}$	80.868	51.308	21.772	95.770	38.763	23.041	41.527	32.399	10.042	16.028
neg. values on test set	0.5 %	0.0 %	1.1 %	9.0%	0.0 %	3.8%	3.0%	1.1 %	1.0 %	0.7 %
$\sum_{w\in\Sigma^*} \hat{p}(w)$	1.007	1.005	0.889	1.075	0.969	0.897	0.978	1.453	0.831	0.970
PAutomaC ranking $\hat{A}^{(k^*)}$	*Last*	*Last*	*Last*	*Last*	*Last*	*Last*	*Last*	*Last*	*Last*	*Last*

Table 1: Results given by the spectral algorithm applied on 10 PAutomaC problems.

Problem number	4	7	12	20	24	30	31	33	39	42
(k^*, m^*)	11, 2	12, 1	13, 3	9, 7	5, 7	9, 5	12, 3	4, 7	7, 1	7, 0
perplexity $L(C_{m^*}^{(k^*)})$	81.408	**51.251**	**21.700**	184.459	**38.736**	23.459	42.710	**32.050**	**10.003**	**16.006**
sum of neg. weights in $B_{m^*}^{(k^*)}$	0.821	0.012	1.850	4.728	0.007	2.566	6.276	0.379	3.500	0.394
PAutomaC ranking $C_{m^*}^{(k^*)}$	*Last*	2^{nd}	4^{th}	*Last*	3^{rd}	*Last*	*Last*	*Last*	2^{nd}	2^{nd}

Table 2: Results given by the method described in Section 3.2 applied on the same PAutomaC problems.

For 6 problems out of the 10 studied, the method significantly improves the quality of the solution allowing 3 of them to be on the PAutomaC podium. Also, note that 3 of the problems that could not be improved were originally the ones giving the most negative values on the test set and that 2 of them where the ones with the highest remaining negative values in the prefix representation before applying on it a normalization procedure. This tends to confirm that the method proposed in this section is able to give good results when the quality of the initial model is close enough to the target solution.

4. Grammatical inference and Structured Low-Rank Approximation

In the previous section, we showed that when the spectral method is able to output a model close to the target, we can improve it by applying a likelihood maximization procedure. Having a good approximation of the target is then crucial and in this section we propose to consider *Structured Low-Rank Approximation* (SLRA) (Markovsky, 2008, 2012) methods for improving the output given by the spectral algorithm. Indeed, the core step of the spectral algorithm is based on a low-rank approximation of the empirical matrix H_S given by a rank-k truncated singular values decomposition : $L_k D_k R_k^T = H_{Sk}$. Note that H_{Sk} is the best rank-k approximation of H_S in term of Frobenius norm (Stewart, 1992). However, H_{Sk} is no longer a Hankel structured matrix. As a consequence, the reconstruction step of the spectral algorithm – building from H_{Sk} the rank-k linear representation $\langle I, (M_x)_{x\in\Sigma}, T\rangle$ defining the series $\hat{p}$ – can be seen as another matrix approximation between the unstructured H_{Sk} and the expected Hankel structured $H_{\hat{p}}$. See Figure 5 for an illustration.

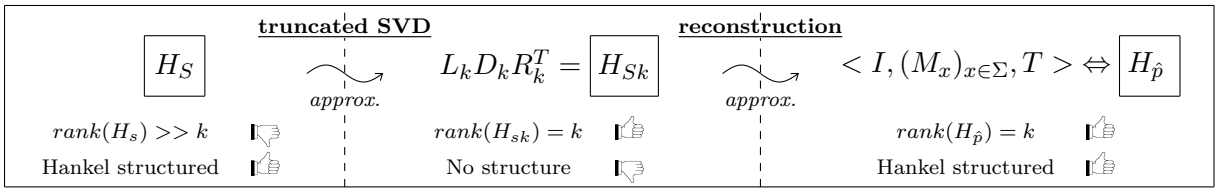


Figure 5: Diagram summarizing the different approximation steps made by the spectral algorithm : The first one is done for obtaining the target rank without the correct matrix structure and the second one can be seen as a recovery the Hankel structure property.

SLRA corresponds to the problem of finding a low rank structure-preserving approximation of a given data matrix. In particular, the SLRA framework supports Hankel structured matrices. Let H_{slra} be a low-rank approximation of H_S obtained by SLRA. Since both rank and structure are correct, we can directly apply the procedure presented in Section 2.2 on H_{slra} to build a rank-k linear representation computing the series $\hat{p}$ without an additional approximation, *i.e.* for any sets U and V indexing H_S we have $\forall u \in U$ and $\forall v \in V$, $H_{slra}[u, v] = \hat{p}(uv)$.

In the next subsection, we introduce the SLRA framework in the context of our grammatical inference problem and evaluate its relevance on a small example. Basically, the SLRA framework can provide, in theory, a nice solution to the problem addressed by spectral methods. However, this framework is strongly non-convex and thus cannot be applied directly on large real problems. To deal with this drawback, we discuss, in a second step, some strategies to deal with big matrices and show some preliminary results on the PAutomaC challenge.

4.1. From grammatical inference to SLRA

Let S be a sample of strings *i.i.d.* from an unknown distribution p of rank k and $H_S \in \mathbb{R}^{m\times n}$ be the empirical Hankel matrix built from S and indexed by a pair $(U, V) \subseteq \Sigma^* \times \Sigma^*$. Let $\mathbf{p_S} = [p_S(w_1), \ldots, p_S(w_{n_p})] \in \mathbb{R}^{n_p}$ be the vector that contains the empirical probabilities of each string w_i of S without duplicates. The Hankel structure of H_S can be modeled from $\mathbf{p_S}$ by an affine structure defined by a mapping $\mathscr{S} : \mathbb{R}^{n_p} \to \mathbb{R}^{m\times n}$ such that :

$$H_s = \mathscr{S}(\mathbf{p_S}) = \sum_{i=1}^{n_p} S_i \, p_S(w_i),$$

where matrices $S_i \in \mathbb{R}^{m\times n}$, indexed by the same pair (U,V), encode the Hankel structure such that $\forall u \in U$ and $\forall v \in V$, $S_i[u, v] = \begin{cases} 1 \text{ if } w_i = uv \\ 0 \text{ otherwise.} \end{cases}$

For example, considering the 3 strings sample $S = \{\epsilon, \epsilon, a\}$ and taking $U = V = \{\epsilon, a\}$, we have $H_S = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \times p_S(\epsilon) + \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \times p_S(a) = \begin{bmatrix} 2/3 & 1/3 \\ 1/3 & 0 \end{bmatrix}$.

The SLRA outputs a Hankel structured matrix $\mathscr{S}(\hat{\mathbf{p}}_\mathbf{S})$ that is a rank-k approximation of H_S by means of the following minimization problem :

$$\underset{\hat{\mathbf{p}}_\mathbf{S} \in \mathbb{R}^{n_p}}{\text{minimize}} \; ||\hat{\mathbf{p}}_\mathbf{S} - \mathbf{p}_\mathbf{S}|| \text{ subject to } rank(\mathscr{S}(\hat{\mathbf{p}}_\mathbf{S})) \leq r. \tag{1}$$

The rank constraint $rank(\mathscr{S}(\hat{\mathbf{p}}_\mathbf{S})) \leq r$ is actually equivalent to find a matrix $R \in \mathbb{R}^{d \times m}$, where $d = m - r$, such that $rank(R) = d$ and $R\mathscr{S}(\hat{\mathbf{p}}_\mathbf{S}) = 0$ (Usevich and Markovsky, 2013). Then, Problem (1) can be reformulated as a double minimization problem as follows :

$$\underset{R:rank(R)=d}{minimize} \; f(R), \text{ where} \tag{2}$$

$$f(R) := min||\hat{\mathbf{p}}_\mathbf{S} - \mathbf{p}_\mathbf{S}||_2^2 \quad \text{subject to} \quad R\mathscr{S}(\hat{\mathbf{p}}_\mathbf{S}) = 0 \tag{3}$$

The inner minimization problem (3) is a least-norm problem and has a closed form solution. On the other hand, the outer problem (2) is by nature non convex making thus the SLRA formulation intractable on large problems. As shown below, the SLRA formulation applied on small problems is very appropriate for the probabilistic grammatical inference problem we consider. However, if some progress has been done recently (Markovsky and Usevich, 2014), this non-convexity issue makes impossible to apply the SLRA approach on full empirical Hankel matrices. We provide at the end of the section a first attempt to deal with this drawback accompanied with an evaluation on PAutomaC.

4.2. Evaluation on a small problem

To evaluate the SLRA approach, we consider the rank-k linear representation of Figure 1, denoted by A, computing the series p and let S be a set of strings i.i.d. from p. In order to stand in a realizable case for SLRA, we consider a small sub-matrix H'_S of H_S following the approach from (Balle et al., 2012) to learn a model. The rank of the target model is $k = 2$ so for this example, this approach only needs the informations from a sub-matrix $H'_S \in \mathbb{R}^{5 \times 5}$ to be applied. Let d_{quad} denote the quadratic distance between two series, define by $d_{quad}(p_1, p_2) = \left[\sum_{w \in \Sigma^*}(p_1(w) - p_2(w))^2\right]^{\frac{1}{2}}$. In order to evaluate the two methods, we used the following experimental setup. First, we consider different sizes of training set i.i.d. from p (from 500 to 50000 strings). For each sample, we fill the matrix H'_S and apply the SLRA optimization problem to obtain a rank-2 approximation H'_{slra}, from which we build a linear representation of the series $\hat{p}_{slra}$. We compare this approach to the method consisting in applying the spectral method on H'_S giving the series $\hat{p}_S$. We repeat each experiment 100 times and we compare the quality of the different methods according to the following 2 measures :

— $d_1(H'_S, H'_{slra}) = ||H_p - H'_S||_2 - ||H_p - H'_{slra}||_2$,
— $d_2(\hat{p}_S, \hat{p}_{slra}) = d_{quad}(\hat{p}_S, p) - d_{quad}(\hat{p}_{slra}, p)$.

The first one evaluates the quality of the low rank approximation with respect to the target matrix H_p and the second one the quality of the inferred model in terms of quadratic distance to the target. A positive value indicates that the SLRA approach is better than the spectral one. The results are reported on Figure 6.

The two evaluation measures confirm that the solution found by the SLRA method is significantly better than the spectral approach. Moreover, using a one-tail Student t-test, the

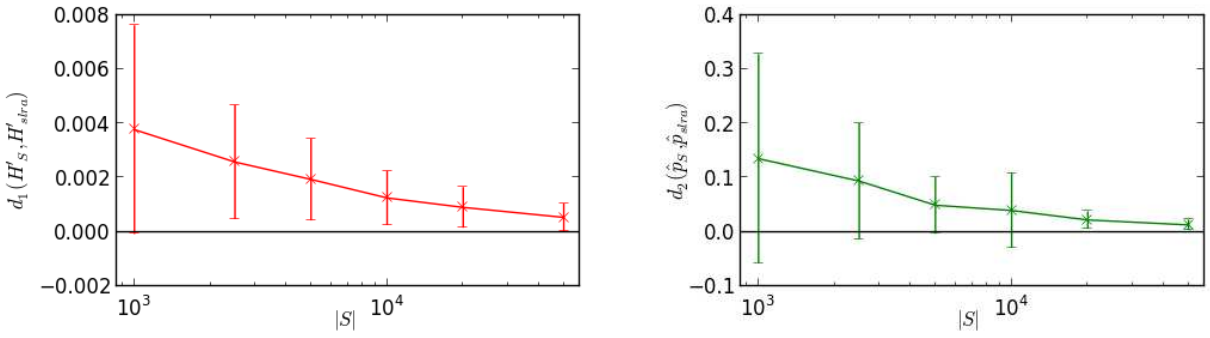


Figure 6: Comparison of the performances of the SLRA approach and the spectral one on the toy problem of Figure 1 according to different learning sample sizes. The results evaluated by the gain in terms of distance d_1 (2-norm) are presented on the left and those in terms of distance d_3 (quadratic distance to the target) are given on the right.

results are statistically significant with a p-value always lower than 10^{-6}. This illustrates that SLRA is clearly able to provide good approximations to a target model when the problem is feasible. In the next section, we present a solution for larger problems.

4.3. Using SLRA on large problems

Due to its non-convexity, the SLRA formulation cannot be applied to real problems : If the matrix is big the problem is intractable and on the other hand if the matrix is too small we may face to a dramatic loss of information and even to an incapacity to obtain a sufficient rank, implying poor results. Another strategy is to use SLRA as a preprocess to improve the quality of the information in the empirical Hankel matrix before applying the spectral algorithm. The simple idea proposed here is to apply SLRA on a submatrix of the empirical Hankel matrix and to apply the spectral method on the Hankel matrix updated with the values returned by SLRA. Given a empirical matrix H_S and a target rank, we propose the following procedure :

- Extract a small squared sub-matrix H'_S from H_S in order to maximize the quantity of information in H'_S such that the rank of H'_S is greater than the target rank and that the size is compatible with the computing resources in order to apply the SLRA problem.
- Apply the SLRA formulation on H'_S with the desired rank to obtain H'_{slra}.
- Replace in the whole matrix H_S each occurrence of the variables in H'_S by their new values in H'_{slra} to obtain a matrix H''_S.
- Apply the spectral algorithm on H''_S.

We have evaluated this strategy on 10 problems of the PAutomaC challenge and compared it to the classical spectral learning algorithm.

We used the following measures to assess the quality of the results : (i) the distance of the Hankel matrices from the target matrix H'_p in terms of Frobenius norm ; (ii) the

Problem	4	7	12	20	24	30	31	33	39	42
dimension H'_S	18×18	18×18	27×27	26×26	12×12	20×20	22×22	12×12	23×23	18×18
$\|\|H'_S - H'_p\|\|_F$	$3.70e^{-3}$	$5.98e^{-3}$	$4.60e^{-3}$	$1.54e^{-3}$	$8.68e^{-3}$	$3.77e^{-3}$	$5.29e^{-3}$	$2.43e^{-3}$	$5.19e^{-3}$	$4.47e^{-3}$
$\|\|H'_{slra} - H'_p\|\|_F$	$\mathbf{3.30e^{-3}}$	$5.98e^{-3}$	$4.60e^{-3}$	$\mathbf{1.47e^{-3}}$	$\mathbf{8.24e^{-3}}$	$\mathbf{3.76e^{-3}}$	$\mathbf{5.25e^{-3}}$	$\mathbf{2.19e^{-3}}$	$\mathbf{5.16e^{-3}}$	$\mathbf{4.32e^{-3}}$
$d_{quad}(\hat{p}_S, p)$	$2.24e^{-3}$	$5.27e^{-3}$	$4.07e^{-3}$	$1.65e^{-3}$	$5.05e^{-3}$	$4.36e^{-3}$	$4.17e^{-3}$	$2.33e^{-3}$	$3.98e^{-3}$	$3.96e^{-3}$
$d_{quad}(\hat{p}_{slra}, p)$	$\mathbf{2.08e^{-3}}$	$5.28e^{-3}$	$4.07e^{-3}$	$\mathbf{1.62e^{-3}}$	$\mathbf{4.77e^{-3}}$	$4.36e^{-3}$	$\mathbf{4.16e^{-3}}$	$\mathbf{2.06e^{-3}}$	$\mathbf{3.97e^{-3}}$	$\mathbf{3.92e^{-3}}$
Perplexity $\hat{p}_S$	80.862	51.308	21.929	95.466	38.763	23.041	41.527	32.395	10.030	16.026
Perplexity $\hat{p}_{slra}$	**80.861**	51.308	**21.927**	**95.457**	**38.756**	23.041	41.529	**32.385**	10.030	16.026

Table 3: Evaluation of SLRA used as a preprocess of the spectral method. The first line indicates the PAutomaC problems considered and the second the selected size of the sub-matrix on which the SLRA is applied. The following two lines shows the modifications made by SLRA on the Hankel matrices in term of $||\cdot||_F$. The next four lines show the modifications made by SLRA on the learned models in terms of d_{quad} and *perplexity*. If SLRA has made an improvement over the classical method, the result is shown in **bold**.

quadratic distance of the learn series with the target series p; (iii) the *perplexity* of the learn models over the test set. The results are reported on Table 3.

First, we observe that, in terms of Frobenius norm, the SLRA-based approach is able to improve the quality of the information contained in the matrix H'_S for 9 problems out of 10. Moreover, for 7 problems, models inferred from H''_S are better in term of quadratic distance than models inferred from H'_S which means that we were able to move closer to the target and a gain in terms of perplexity is observed for 5 problems. This shows that SLRA can bring some valuable information to the empirical matrix which leads to produce better models. As a conclusion, these preliminary results show that SLRA can be a very relevant tool for learning low-rank linear representations.

5. Conclusion

In this paper, we have studied two possible directions for improving the solution output by spectral methods for learning linear representation of stochastic languages.

First, we provided an approach allowing one to maximize the likelihood of a sample from a model given by the spectral methods. This strategy is based on a renormalization of the model accompanied with an iterative procedure allowing one to reduce the magnitude of negative coefficients thanks to an extension of the model. Once the negative coefficients are eliminated, we obtain a starting point for the BW algorithm used afterwards to update the coefficients of the model in order to maximize the likelihood of the learning sample. Our experiments show that this approach can improve the spectral algorithm if the model output by this algorithm is sufficiently close to the target solution.

In a second step, we studied the *Structured Low Rank Approximation* methods in the context of our learning problem. This approach allows one to obtain a low-rank representation of a matrix while preserving its (Hankel) structure which corresponds exactly to the problem spectral methods aims at solving. We have illustrated the interest of this approach on a synthetic problem and its limitations on big Hankel matrices – coming with real applications– due to the non-convexity of its nature. We have proposed a preliminary

approach consisting of applying the SLRA framework on a small feasible subpart of the full Hankel matrix. This strategy tends to improve the final output of the method but is not completely convincing since the SLRA is only modifying a relatively small amount of data of the Hankel matrix H_S. As a consequence, the model obtained is still relatively close to the spectral solution and in our preliminary experiments, combining SLRA with the Baum-Welch procedure does not bring a significative gain.

A natural perspective of this work is to find new approaches for applying SLRA on a larger amount of data of the empirical matrix. For example, trying to use many SLRA on random sub-matrices or projections with a relevant combination of the results is a natural perspective.

Références

A. Anandkumar, D.P. Foster, D. Hsu, S. Kakade, and Y.-K. Liu. A spectral algorithm for latent dirichlet allocation. In *Proceedings of NIPS*, pages 926–934, 2012.

R. Bailly. *Méthodes spectrales pour l'inférence grammaticale probabiliste de langages stochastiques rationnels*. PhD thesis, Aix-Marseille Université, 2011.

R. Bailly and F. Denis. Absolute convergence of rational series is semi-decidable. *Inf. Comput.*, 209(3) :280–295, 2011.

R. Bailly, F. Denis, and L. Ralaivola. Grammatical inference as a principal component analysis problem. In *Proceedings of ICML*, 2009.

R. Bailly, A. Habrard, and F. Denis. A spectral approach for probabilistic grammatical inference on trees. In *Proceedings of ALT*, pages 74–88, 2010.

B. Balle and M. Mohri. Spectral learning of general weighted automata via constrained matrix completion. In *Proceedings of NIPS*, pages 2168–2176, 2012.

B. Balle, A. Quattoni, and X. Carreras. A spectral learning algorithm for finite state transducers. In *Proceedings of ECML/PKDD (1)*, pages 156–171, 2011.

B. Balle, A. Quattoni, and X. Carreras. Local loss optimization in operator models : A new insight into spectral learning. In *Proceedings of ICML*, 2012.

B. Balle, W. Hamilton, and J. Pineau. Methods of moments for learning stochastic languages : Unified presentation and empirical comparison. In *Proceedings of ICML*, 2014.

S. B. Cohen, K. Stratos, M. Collins, D. P. Foster, and L. Ungar. Experiments with spectral learning of latent-variable PCFGs. In *Proceedings of NAACL*, 2013.

F. Denis and Y. Esposito. On rational stochastic languages. *Fundam. Inform.*, 86(1-2) : 41–77, 2008.

F. Denis, M. Gybels, and A. Habrard. Dimension-free concentration bounds on hankel matrices for spectral learning. In *Proceedings of ICML*, 2014.

S. W. Dharmadhikari. Sufficient conditions for a stationary process to be a function of a finite markov chain. *Ann. Math. Statist.*, pages 1033–1041, 1963.

D. Hsu, S.M. Kakade, and T. Zhang. A spectral algorithm for learning hidden markov models. In *Proceedings of COLT*, 2009.

A. Kulesza, N. Raj Rao, and S. Singh. Low-rank spectral learning. In *Proceedings of the 17th Conference on Artificial Intelligence and Statistics*, 2014.

F.M. Luque, A. Quattoni, B. Balle, and X. Carreras. Spectral learning for non-deterministic dependency parsing. In *Proceedings of EACL*, pages 409–419, 2012.

I. Markovsky. *Low Rank Approximation : Algorithms, Implementation, Applications*. Communications and Control Engineering. Springer, 2012.

I. Markovsky and K. Usevich. Software for weighted structured low-rank approximation. *J. Comput. Appl. Math.*, 256 :278–292, 2014.

Ivan Markovsky. Structured low-rank approximation and its applications. *Automatica*, 44 (4) :891–909, 2008.

A.P. Parikh, L. Song, and E.P. Xing. A spectral algorithm for latent tree graphical models. In *Proceedings of ICML*, pages 1065–1072, 2011.

G. W. Stewart. On the early history of the singular value decomposition, 1992.

K. Usevich and I. Markovsky. Variable projection for affinely structured low-rank approximation in weighted 2-norms. *J. Comput. Appl. Math.*, 2013. doi : 10.1016/j.cam.2013.04.034. URL http://arxiv.org/abs/1211.3938.

S. Verwer, R. Eyraud, and C. de la Higuera. Results of the pautomac probabilistic automaton learning competition. *Journal of Machine Learning Research - Proceedings Track*, 21 :243–248, 2012.